Applied Soft Computing 57 (2017) 605–614

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

k-mw-modes: An algorithm for clustering categorical matrix-object data

Fuyuan Cao[a], Liqin Yu[a], Joshua Zhexue Huang[b], Jiye Liang[a,*]

[a] Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, School of Computer and Information Technology, Shanxi University, Taiyuan 030006, China
[b] College of Computer Sciences & Software Engineering, Shenzhen University, Shenzhen 518060, China

ARTICLE INFO

Article history:
Received 13 August 2016
Received in revised form 4 April 2017
Accepted 11 April 2017
Available online 20 April 2017

Keywords:
Categorical data
Matrix-object
k-mw-modes algorithm

ABSTRACT

In data mining, the input of most algorithms is a set of *n* objects and each object is described by a feature vector. However, in many real database applications, an object is described by more than one feature vector. In this paper, we call an object described by more than one feature vector as a matrix-object and a data set consisting of matrix-objects as a matrix-object data set. We propose a *k*-multi-weighted-modes (abbr. *k*-mw-modes) algorithm for clustering categorical matrix-object data. In this algorithm, we define the distance between two categorical matrix-objects and a multi-weighted-modes representation of cluster prototypes is proposed. We give a heuristic method to choose the locally optimal multi-weighted-modes in the iteration of the *k*-mw-modes algorithm. We validated the effectiveness and benefits of the *k*-mw-modes algorithm on the five real data sets from different applications.

1. Introduction

In data mining, the input of an algorithm in most cases is a data set *X*, also called a table or matrix. The data set consists of *n* objects $\{x_1, x_2, \ldots, x_n\}$ and each object is described by *m* attributes $\{A_1, A_2, \ldots, A_m\}$ [1]. Most importantly, each object in *X* only corresponds with a feature vector $(x_{i1};x_{i2};\ldots;x_{im})$, $i \in \{1, 2, \ldots, n\}$. However, in many real applications, a database often contains multiple tables. There are one-one, one-many or many-many relationships between two tables. Thus, an object usually corresponds with more than one transactional record. A real database application example from http://www.taobao.com is described in Table 1.

There are two parts in Table 1. The left half describes the basic information of users and the right one records that each user visited different brands in different time points, where the attribute *Visited_Times* represents the visiting-times of a user on the same day for one brand. We call the left part as a master table and the right one as a detail table in database. Therefore, two parts in Table 1 exist a typical one-many relationship. Data in Table 1 have the following characteristics:

- **Correlation**: Data from the master table and the detail table maybe have some correlations. Users with different sex or age maybe have different preferences. For example, the female user of 24 years old from Table 1 visited the commodities that are usually used by most female users, such as *JOSINY* and *WETHERM*. However, the female user of 40 years old visited the commodities used by men or women, maybe because she needs to take after their families.
- **One-many**: Each user in the master table corresponds with more than one record in the detail table. Moreover, the number of brands visited by different users is often different in Table 1. For example, the user 10944750 has 11 records while the user 8149250 has 4 records.
- **Mixed**: In most cases, an object is described by categorical and numerical attributes together. For example, in the detail table, Brand_Name is a categorical attribute while Visited_Times is a numerical attribute.
- **Evolution**: Some attribute values will change as time goes on. For example, a user visits one brand repeatedly in this month, but the brand may be not visited by him or her in the next month. In other words, the change of a user's behavior is a dynamic evolution process with time.

From the detail table, we can see clearly that every user visited one brand at least and a brand may be browsed by many users. Besides, a brand may be visited several times by a user in a day.

[*] Corresponding author.
E-mail addresses: cfy@sxu.edu.cn (F. Cao), 1367545521@qq.com (L. Yu), zx.huang@szu.edu.cn (J.Z. Huang), ljy@sxu.edu.cn (J. Liang).

http://dx.doi.org/10.1016/j.asoc.2017.04.019
1568-4946/© 2017 Elsevier B.V. All rights reserved.

Table 1
A real application example from Taobao.

User_ID	*User_Sex*	*Age*	*Brand_ID*	*Brand_Name*	*Visit_Date*	*Visited_Times*
10944750	Female	24	13451	WETHERM	06-04	8
			21110	SEMIR	06-07	1
			25687	JOSINY	05-08	11
			25687	JOSINY	05-15	4
			25687	JOSINY	05-17	2
			25687	JOSINY	06-06	2
			25687	JOSINY	06-15	2
			25687	JOSINY	07-02	5
			25687	JOSINY	07-25	2
			25687	JOSINY	08-09	3
			25687	JOSINY	08-13	1
8149250	Male	29	18805	YEARCON	05-25	1
			18805	YEARCON	05-26	4
			18805	YEARCON	06-12	1
			21110	SEMIR	06-30	1
1832000	Female	40	25687	JOSINY	08-14	18
			25687	JOSINY	08-15	3
			25687	JOSINY	06-26	5
			25687	JOSINY	08-11	1
			10151	OBERORA	08-03	1
			18805	YEARCON	07-26	5

Of course, it may also be visited many times by a user in several days. Obviously, if a user visited many times about a brand, he or she may be interested in this commodity. For example, for the user 10944750, the *JOSINY* is visited in continuous four months, and there are several visiting times in every month. So, we can predict the user is likely very fond of the *JOSINY*. However, the *SEMIR* is visited only once by the user 10944750 in this data set, by which we know that the user may have less like about it compared with the *JOSINY*. Such a data representation shown in Table 1 is widespread in banking, insurance, telecommunication, retails, and medical databases. Therefore, it is necessary to develop a method that can discover user groups with different behavior patterns from the detail table instead of the master table. Because the behavior analysis can help managers obtain more valuable information for decision making.

Clustering is a widely used method to find different user groups in real applications [2] and the master table tends to be taken as its input. But the information in the master table cannot enough reflect the behavior characteristics of a user. More importantly, in traditional clustering algorithms, the dissimilarity measure between two objects is based on the value difference of two feature vectors. For the detail table, each user has more than one transactional record. In other words, each user is described by multiple feature vectors. Therefore, some classical dissimilarity measures, such as Euclidean distance, Manhattan distance and Hamming distance, cannot be used to process this kind of data directly.

In the detail table, each user has multiple feature vectors, each of which is described by numerical and categorical attributes together in most cases. How to define a dissimilarity measure between two users is a very crucial problem, because it has direct effects on clustering results. For simplicity, in this paper, we only investigate the clustering algorithm for the detail table whose each record is described by categorical attributes. The k-modes algorithm [3] has realized the clustering of the categorical data sets compared with the k-means algorithm [4], but it still has some shortcomings. Only the data sets whose each object only contains one record can be clustered by the k-modes algorithm. Obviously, if the problem above wants to be solved with the k-modes algorithm, the data sets need to be compressed as the form that the algorithm required by selecting an attribute value whose frequency is the highest. Thus, lots of information is at a loss in the data so that the clustering results are unfaithful.

Without loss of generality, a general description of detail information in Table 1 is illustrated as follows. Suppose that $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ is a set of n objects described by m attributes $\{A_1, A_2, \ldots, A_m\}$, where $X_i = (X_{i1};X_{i2};\ldots;X_{im})$ and $X_{is} = [v_{i1s}, v_{i2s}, \ldots, v_{ir_is}]'$.

r_i represents the number of records in X_i and v_{ijs} denotes the jth value of X_i on A_s. We call X_i as a matrix-object and $\mathbf{X}$ as a matrix-object data set. Suppose that V^s represents the domain values of the attribute A_s in $\mathbf{X}$ and $V_{X_i}^{A_s}$ denotes a set of values on the attribute A_s for X_i. Obviously, $\bigcup_{i=1}^{n} V_{X_i}^{A_s} = V^s$. In traditional data representation, an object is only described by a feature vector or a record while a matrix-object is usually represented by multiple feature vectors or records. Therefore, a matrix-object is a general representation of a traditional object.

In this paper, we propose a new clustering algorithm, the k-mw-modes algorithm, to cluster categorical matrix-object data. The main contributions are summarized as follows:

- We define a new dissimilarity measure to calculate the distance between two categorical matrix-objects.
- We give a new representation and update way of the cluster centers to optimize the clustering process.
- We give a heuristic method to choose the cluster center of a set.
- We propose the k-mw-modes clustering algorithm to cluster categorical matrix-object data.
- Experimental results on the real data sets have shown the effectiveness of the k-mw-modes algorithm.

The rest of this paper is organized as follows. In Section 2, we propose the k-mw-modes algorithm. In Section 3, we give a heuristic method to choose the locally optimal multi-weighted-modes for the k-mw-modes algorithm. In Section 4, we show experimental results on the five real data sets from different applications. In Section 5, we review some related work. We give conclusions and future work in Section 6.

2. *k*-multi-weighted-modes clustering

The k-modes clustering algorithm consists of three components: (1) representation of cluster centroids, (2) allocation of objects into clusters and (3) updates of cluster centroids. In this section, we present the k-mw-modes algorithm that uses the k-modes clustering process to cluster categorical matrix-object data. In this algorithm, we define a dissimilarity measure to calculate the distance between two matrix-objects and give a kind of representation and update way of cluster centers.

2.1. Distance between two matrix-objects

Given two matrix-objects X_i and X_j, which are described by m categorical attributes $\{A_1, A_2, \ldots, A_m\}$, the dissimilarity measure between X_i and X_j is defined as

$$d(X_i, X_j) = \frac{1}{2}\sum_{s=1}^{m}\delta(X_{is}, X_{js}) \tag{1}$$

where

$$\delta(X_{is}, X_{js}) = \sum_{v\in V_{X_i}^{A_s}\cup V_{X_j}^{A_s}}\left|\frac{\sum_{p=1}^{r_i} f(v, v_{ips})}{r_i} - \frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j}\right| \tag{2}$$

and

$$f(x, y) = \begin{cases} 1, & \textit{if} \quad x == y. \\ 0, & \textit{otherwise}. \end{cases} \tag{3}$$

Here, $f(\cdot, \cdot)$ is a function whose value is 1 if two parameter values are equal, otherwise its value is 0. $|\cdot|$ represents the absolute value of a value.

In addition, as $0 \le \delta(X_{is}, X_{js}) \le 2$, we add a normalization factor $\frac{1}{2}$ in Eq. (1). We have $\delta(X_{is}, X_{js}) = 2$ when $V_{X_i}^{A_s} \cap V_{X_j}^{A_s} = \emptyset$.

We can prove that the dissimilarity measure $d(X_i, X_j)$ is a distance metric satisfying three properties as follows:

(1) Nonnegativity: $d(X_i, X_j) \ge 0$ and $d(X_i, X_i) = 0$;
(2) Symmetry: $d(X_i, X_j) = d(X_j, X_i)$;
(3) Triangle inequality: $d(X_i, X_j) + d(X_j, X_k) \ge d(X_i, X_k)$.

Obviously, we can easily prove the first two properties because $|\cdot|$ is the symbol of an absolute value. The triangle inequality as the third property is verified as follows.

Proof 1. To prove the inequality $d(X_i, X_j) + d(X_j, X_k) \ge d(X_i, X_k)$, we only need to demonstrate

$$\delta(X_{is}, X_{js}) + \delta(X_{js}, X_{ks}) \ge \delta(X_{is}, X_{ks}), \quad s \in \{1, 2, \ldots, m\}.$$

With Eq. (2), the inequality above can be rewritten as

$$\begin{aligned}
&\sum_{v\in V_{X_i}^{A_s}\cup V_{X_j}^{A_s}}\left|\frac{\sum_{p=1}^{r_i} f(v, v_{ips})}{r_i} - \frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j}\right| + \sum_{v\in V_{X_j}^{A_s}\cup V_{X_k}^{A_s}}\left|\frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j} - \frac{\sum_{l=1}^{r_k} f(v, v_{kls})}{r_k}\right| \\
&= \sum_{v\in V_{X_i}^{A_s}\cup V_{X_j}^{A_s}\cup V_{X_k}^{A_s}}\left|\frac{\sum_{p=1}^{r_i} f(v, v_{ips})}{r_i} - \frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j}\right| + \sum_{v\in V_{X_i}^{A_s}\cup V_{X_j}^{A_s}\cup V_{X_k}^{A_s}}\left|\frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j} - \frac{\sum_{l=1}^{r_k} f(v, v_{kls})}{r_k}\right| \\
&= \sum_{v\in V_{X_i}^{A_s}\cup V_{X_j}^{A_s}\cup V_{X_k}^{A_s}}\left(\left|\frac{\sum_{p=1}^{r_i} f(v, v_{ips})}{r_i} - \frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j}\right| + \left|\frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j} - \frac{\sum_{l=1}^{r_k} f(v, v_{kls})}{r_k}\right|\right) \\
&\ge \sum_{v\in V_{X_i}^{A_s}\cup V_{X_j}^{A_s}\cup V_{X_k}^{A_s}}\left|\frac{\sum_{p=1}^{r_i} f(v, v_{ips})}{r_i} - \frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j} + \frac{\sum_{q=1}^{r_j} f(v, v_{jqs})}{r_j} - \frac{\sum_{l=1}^{r_k} f(v, v_{kls})}{r_k}\right| \\
&= \sum_{v\in V_{X_i}^{A_s}\cup V_{X_j}^{A_s}\cup V_{X_k}^{A_s}}\left|\frac{\sum_{p=1}^{r_i} f(v, v_{ips})}{r_i} - \frac{\sum_{l=1}^{r_k} f(v, v_{kls})}{r_k}\right| \\
&= \sum_{v\in V_{X_i}^{A_s}\cup V_{X_k}^{A_s}}\left|\frac{\sum_{p=1}^{r_i} f(v, v_{ips})}{r_i} - \frac{\sum_{l=1}^{r_k} f(v, v_{kls})}{r_k}\right|.
\end{aligned}$$

□

Table 2
An example of two matrix-objects.

ID	A_1	A_2
X_1	2	4
	2	5
	3	4
	4	4
	4	2
X_2	2	4
	4	2
	5	2
	5	1

The above proof verifies that the triangle inequality property holds on one attribute. It naturally can be generalized to multiple attributes. It follows that we have $d(X_i, X_j) + d(X_j, X_k) \ge d(X_i, X_k)$. Therefore, the dissimilarity measure $d(\cdot, \cdot)$ is a distance metric.

Next, we give an example of calculating the distance between two matrix-objects as follows.

Example 1. Given two matrix-objects described by two categorical attributes whose values are represented by integers. The details are illustrated in Table 2.

According to Eq. (1), the distance between X_1 and X_2 on the attribute A_1 can be computed as $|\frac{2}{5} - \frac{1}{4}| + |\frac{1}{5} - \frac{0}{4}| + |\frac{2}{5} - \frac{1}{4}| + |\frac{0}{5} - \frac{2}{4}| = 1$. Similarly, the dissimilarity measure is $\frac{11}{10}$ on the attribute A_2. Therefore we have $d(X_1, X_2) = (1 + \frac{11}{10})/2 = \frac{21}{20}$.

The dissimilarity measure $d(\cdot, \cdot)$ is also a generalization of the simple matching dissimilarity measure that is used in the k-modes algorithm. In other words, if two matrix-objects have only one record respectively, their distance can be calculated by Eq. (1) as well. For instance, suppose that there are two objects X_1, X_2 described by four attributes A_1, A_2, A_3, A_4, $X_1 = (a;c;d;c)$, $X_2 = (a;d;d;b)$. The distance between them by Eq. (1) is $(|1-1| + (|1-0| + |0-1|) + |1-1| + (|1-0| + |0-1|))/2 = 2$, and in the simple matching dissimilarity measure, it is $0 + 1 + 0 + 1 = 2$. So, the dissimilarity measure in the k-modes algorithm is a special case of $d(\cdot, \cdot)$.

With Eq. (1), we design an algorithm to calculate the distance between two matrix-objects. The details are described in Algorithm 1, which is named *ACDM* (an Algorithm of Calculating Distance between Matrix-objects).

Algorithm 1. The *ACDM*

```
Input: – $X_i, X_j$: two matrix-objects described by $m$ attributes;
Output: – distance: the distance between $X_i$ and $X_j$;
Method:
distance = 0;
for $s = 1 to m$ do
  Obtain $X_{is} = [v_{i1s}, v_{i2s}, \ldots, v_{ir_is}]'$ and $X_{js} = [v_{j1s}, v_{j2s}, \ldots, v_{jr_js}]'$;
  $V = V_{X_i}^{A_s} \cup V_{X_j}^{A_s} = \{v_1^s, v_2^s, \ldots, v_u^s\}$;
  sum = 0;
  for $t = 1 to u$ do
    $s1 = 0, s2 = 0$;
    for $p = 1 tor_i$ do
      if $v_t^s == v_{ips}$
        $s1 = s1 + 1$; then
      end if
    end for
    for $q = 1 tor_j$ do
      if $v_t^s == v_{jqs}$ then
        $s2 = s2 + 1$;
      end if
    end for
    $sum = sum + |s1/r_i - s2/r_j|$;
  end for
  $distance = distance + sum/2$;
end for
return distance;
```

Algorithm 2. The *GAFMWM*

```
Input: – X: A set of $n$ matrix-objects described by $m$ attributes;
Output: – $Q$: The cluster center of X;
Method:
$Q = \emptyset$;
for $s = 1 to m$ do
  Generate a set $q^s = \{q_s^1, q_s^2, \ldots, q_s^{2^{|V^s|}-1}\}$ in $V^s$ by binomial
theorem;
  for $j = 1$ to $2^{|V^s|} - 1$ do
    $Q_{A_s}^j = \emptyset$;
    for $p = 1$ to $|q_s^j|$ do
      $Q_{A_s}^j = Q_{A_s}^j \cup \{(v_p^s, \sum_{i=1}^n f_i'(v_p^s)) | v_p^s \in q_s^j\}$;
    end for
    Computer $F_j = \sum_{i=1}^n \delta(X_{is}, Q_{A_s}^j)$ by Eq. (2);
    if $j = 1$ then
      set minValue $= F_j, Q_{A_s} = Q_{A_s}^j$;
    else if $F_j < minValue$ then
      set minValue $= F_j, Q_{A_s} = Q_{A_s}^j$;
    end if
  end for
  set $Q = Q \cup Q_{A_s}$;
end for
return $Q$;
```

2.2. Multi-weighted-modes as cluster centers

Suppose that **X** is a matrix-object data set described by m categorical attributes, the cluster center of **X** is defined as follows.

Definition 1. Let $V_{X_i}^{A_s} = \{v_1^s, v_2^s, \ldots, v_{u_s'}^s\}$ be the domain values of X_i on the attribute A_s, the frequency of v_u^s $(1 \le u \le u_s')$ in X_i is defined as

$$f_i'(v_u^s) = \sum_{p=1}^{r_i} f(v_u^s, v_{ips}), \tag{4}$$

With Eq. (4), we can easily obtain the frequency of each attribute value in X_i on A_s. Thus, X_{is} can be represented as $X_{is} = \{(v_u^s, f_i'(v_u^s)) | v_u^s \in V_{X_i}^{A_s}\}$ or $X_{is} = \{(v_u^s, \frac{f_i'(v_u^s)}{r_i}) | v_u^s \in V_{X_i}^{A_s}\}$. We call $\frac{f_i'(v_u^s)}{r_i}$ as the weight of v_u^s in X_i.

Definition 2. Let **X** = $\{X_1, X_2, \ldots, X_n\}$ be a set containing n matrix-objects and $Q = (Q_{A_1}; Q_{A_2}; \ldots; Q_{A_m})$ be a matrix-object. They are all described by m categorical attributes. Q is the multi-weight-modes or the center of **X** if Q minimizes the following function

$$F(\mathbf{X}, Q) = \sum_{i=1}^{n} d(X_i, Q), \tag{5}$$

where $X_i \in \mathbf{X}$ and $d(X_i, Q)$ can be calculated by Eq. (1).

To minimize $F(\mathbf{X}, Q)$, we only need to minimize $\sum_{i=1}^n \delta(X_{is}, Q_{A_s})$, the sum of the distance between Q and each matrix-objects in **X** on the attribute A_s where $s \in \{1, 2, \ldots, m\}$. As the attribute values in Q_{A_s} must be from the values in V^s that represents the domain values of **X** on A_s, the number of categorical values in Q_{A_s} is between 1 and $|V^s|$. According to Definition 2, we can compute $\sum_{i=1}^n (f_i'(v))(v \in V^s)$ in **X**. If we choose u_s values $\{v_1^s, v_2^s, \ldots, v_{u_s}^s\}$ from V^s as the values of Q_{A_s}, there are $C_{|V^s|}^{u_s}$ combinations. For a given combination, its component is represented by $(v_j^s, \sum_{i=1}^n (f_i'(v_j^s)))$ where $j \in \{1, 2, \ldots, u_s\}$. It follows that the total number of possible sets for Q_{A_s} is $\sum_{u_s=1}^{|V^s|} C_{|V^s|}^{u_s}$. Therefore, we need to traverse every combination to find a Q_{A_s}, which minimizes $\sum_{i=1}^n \delta(X_{is}, Q_{A_s})$. A global optimization algorithm for finding multi-weighted-modes is described in Algorithm 2, which is named *GAFMWM* (A Global Algorithm of Finding Multi-Weighted-modes).

2.3. The k-mw-modes algorithm

Given Eq. (1) as the distance measure between two matrix-objects, the k-mw-modes algorithm for clustering a matrix-object set **X** = $\{X_1, X_2, \ldots, X_n\}$ into $k(\ll n)$ clusters minimizes the following objective function

$$F'(W, Q) = \sum_{l=1}^{k} \sum_{i=1}^{n} \omega_{li} d(X_i, Q_l),$$

subject to

$$\omega_{li} \in \{0, 1\}, \quad 1 \le l \le k, \quad 1 \le i \le n, \tag{6}$$

$$\sum_{l=1}^{k} \omega_{li} = 1, \quad 1 \le i \le n, \tag{7}$$

$$0 < \sum_{i=1}^{n} \omega_{li} < n, \quad 1 \le l \le k, \tag{8}$$

where $W = [\omega_{li}]$ is a k-by-$n\{0, 1\}$ matrix in which $\omega_{li} = 1$ indicates that object X_i is allocated to cluster l, $Q = [Q_1, Q_2, \ldots, Q_k]$ in which $Q_l \in Q$ is the multi-weighted modes of cluster l.

$F'(W, Q)$ can be solved with an iterative process to solve two subproblems iteratively until the process converges. The first step is to fix $Q = Q^t$ at iteration t and solve the reduced problem $F'(W, Q^t)$ with Eq. (1) to find W^t that minimizes $F'(W, Q^t)$. The second step is to fix W^t and solve the reduced problem $F'(W^t, Q)$ by using algorithm *GAFMWM* to find Q^{t+1} that minimizes $F'(W^t, Q)$. We give the description of the algorithm in Algorithm 3, which is named as the k-mw-modes algorithm.

Algorithm 3. The k-mw-modes algorithm

```
Input:
– X: a data set of $n$ matrix-objects described by $m$ attributes;
– $k$: the number of clusters need to be clustered;
– $\varepsilon$: A threshold;
– idCenters: the id of $k$ initial centers;
Output:
– cid: the labels of all objects after clustering;
– num: the iterations;
Method:
Let $Q$ store the $k$ initial centers by the indexes in idCenters;
value=0, num=0;
while num ≤ 100 do
  newvalue=0;
  for $i = 1$ to $n$ do
    for $j = 1$ to $k$ do
      Calculate the distance between $i$th object and $j$th clustering center by Eq. (1);
    end for
    Arrange the $i$th object to the $l$th cluster if $l = argmin_{j=1}^{k}\{d(X_i, Q_j)\}$.
    newvalue=newvalue+$\min_{j=1}^{k}\{d(X_i, Q_j)\}$;
  end for
  If $|newvalue - value| \leq \varepsilon$, break; Else $value = newvalue$ and $num = num + 1$;
  for $i = 1$ to $k$ do
    Update the cluster centers $Q$ with Algorithm 2;
  end for
end while
```

The computation complexity of the k-mw-modes algorithm is analyzed as follows.

- The computation complexity for calculation of the distance between two matrix-objects on A_s is $\mathcal{O}(|V^s|)$. The computation complexity of the distance between two matrix-objects in m attributes is $\mathcal{O}(m \times |V'|)$, where $|V'| = \max\{|V^s|, 1 \leq s \leq m\}$.
- Updating cluster centers. The main goal of updating cluster centers is to find the multi-weighted-modes in each cluster according to the partition matrix W. The computational complexity for this step is $\mathcal{O}(km \times 2^{|V'|})$, where $|V'| = \max\{|V^s|, 1 \leq s \leq m\}$.

If the clustering process needs t iterations to converge, the total computational complexity of the k-mw-modes algorithm is $\mathcal{O}(nmtk \times 2^{|V'|})$, where $|V'| = \max\{|V^s|, 1 \leq s \leq m\}$. It is obviously that the time complexity of the proposed algorithm increases linearly as the number of objects and clusters increases, and increases exponentially with the increasing of the number of attribute values. The space complexity of the k-mw-modes algorithm is $\mathcal{O}((n + k)\sum_{s=1}^{m}|V^s|)$.

Theorem 1. *The k-mw-modes algorithm converges to a local minimal solution in a finite number of iterations.*

Proof 2. We note that the number of possible values for the center of a cluster is $N = \Pi_{s=1}^{m}\sum_{u_s=1}^{|V^s|}C_{|V^s|}^{u_s}$, where $|V^s|$ is the number of unique values on A_s and $C_{|V^s|}^{u_s}$ is the number of combinations of choosing u_s values from a set of $|V^s|$ values. To divide a data set into k clusters, the number of possible partitions is finite. We can show that each possible partition only occurs once in the clustering process. Let W^h be a partition at iteration h. We can obtain Q^h that depends on W^h.

Suppose that $W^{h_1} = W^{h_2}$, where h_1 and h_2 are two different iterations, i.e., $h_1 \neq h_2$. If Q^{h_1} and Q^{h_2} are obtained from W^{h_1} and W^{h_2}, respectively, then $Q^{h_1} = Q^{h_2}$ since $W^{h_1} = W^{h_2}$. Therefore, we have

$$F'(W^{h_1}, Q^{h_1}) = F'(W^{h_2}, Q^{h_2}).$$

However, the value of the objective function $F'(\cdot, \cdot)$ generated by the k-MW-modes algorithm is strictly decreasing. h_1 and h_2 must be two consecutive iterations in which the clustering result is no longer change and the clustering process converges. Therefore, the k-mw-modes algorithm converges in a finite number of iterations. □

3. A heuristic method for updating cluster centers

The *GAFMWM* for finding cluster centers is not efficient if the number of domain values is very large. In this section, we give a heuristic method of updating cluster centers in the k-mw-modes clustering process. For X_i, $X_j \in \mathbf{X}$, we have $V_{X_i}^{A_s} = V_{X_j}^{A_s}$ or $V_{X_i}^{A_s} \neq V_{X_j}^{A_s}$ on the attribute A_s. Even if $V_{X_i}^{A_s} = V_{X_j}^{A_s}$, the frequency of the same attribute value may be different in X_i and X_j, because a value maybe appears more than once in a given matrix-object. The higher the frequency of a value in a given matrix-object is, the more the possibility of the value as the cluster center is. In order to describe the possibility of a value as the cluster center, the weight of a value is defined as follows.

Definition 3. Let $V^s = \{v_1^s, v_2^s, \ldots, v_{u'_s}^s\}$ be the domain values of $\mathbf{X}$ on the attribute A_s. For any $u \in \{1, 2, \ldots, u'_s\}$, the weight of v_u^s in $\mathbf{X}$ is defined as

$$\omega(v_u^s) = \frac{1}{n}\sum_{i=1}^{n}\frac{f_i'(v_u^s)}{r_i}. \tag{9}$$

Obviously, we can obtain the weight of each value in V^s with Eq. (9) and arrange them in the descending order of the weight. Suppose that $V^s = \{v_{q_1}^s, v_{q_2}^s, \ldots, v_{q_{u'_s}}^s\}$ and $\omega(v_{q_1}^s) \geq \omega(v_{q_2}^s) \geq \cdots \geq \omega(v_{q_{u'_s}}^s)$. According to Definition 2, we have $\sum_{i=1}^{n}f_i'(v_{q_1}^s), \sum_{i=1}^{n}f_i'(v_{q_2}^s), \cdots, \sum_{i=1}^{n}f_i'(v_{q_{u'_s}}^s)$. If there are $u_s(1 \leq u_s \leq u'_s)$ values in Q_{A_s}, $Q_{A_s} = \{(v_j^s, \sum_{i=1}^{n}f_i'(v_j^s))|q_1 \leq j \leq q_{u_s}\}$. Here, in order to increase efficiency, we set $u_s = round(\frac{\sum_{i=1}^{n}|V_{X_i}^{A_s}|}{n})$ for Q_{A_s}. By the way above, the centers on other attributes can be found as well. The heuristic algorithm is described in Algorithm 4, which is called *HAFMWM* (A Heuristic Algorithm of Finding Multi-Weighted-modes).

Algorithm 4. The *HAFMWM*

```
Input: – X: A set of $n$ matrix-objects described by $m$ attributes;
Output: – $Q$: A cluster center of X;
Method:
for $s = 1$ to $m$ do
  sum=0, $Q_{A_s} = \emptyset$;
  for $i = 1$ to $n$ do
    $sum = sum + |V_{X_i}^{A_s}|$;
  end for
  $u_s = round(sum/n)$;
  for $i = 1$ to $|V^s|$ do
    Calculate the weight $\omega(v_i^s)$ of $v_i^s \in V^s$ by Eq. (9);
  end for
  Arrange $\{v_1^s, v_2^s, \cdots, v_{|V^s|}^s\}$ in the descending order of their weight;
  for $p = 1$ to $u_s$ do
    Compute $\sum_{i=1}^{n}f_i'(v_p^s)$;
    $Q_{A_s} = Q_{A_s} \cup \{(v_p^s, \sum_{i=1}^{n}f_i'(v_p^s))\}$;
  end for
  $Q = Q \cup Q_{A_s}$;
end for
return $Q$;
```

The time complexity of *HAFMWM* is $\mathcal{O}(km|V'|)$, where $|V'| = \max\{|V^s|, 1 \leq s \leq m\}$. Obviously, the computation complexity of *HAFMWM* is less than *GAFMWM*.

Next, we give an example to describe the process of finding cluster centers by *HAFMWM* as follows.

Example 2. Suppose that $\mathbf{X} = \{X_1, X_2, X_3, X_4\}$ is a set described by one categorical attribute A_1. X_1, X_2, X_3, X_4 are matrix-objects and contain five, four, four and two records, respectively. The details are described in Table 3.

Table 3
An example data set for finding center.

ID	A_1	ID	A_1	ID	A_1	ID	A_1
X_1	3	X_2	3	X_3	3	X_4	3
	3		6		5		7
	5		4		6		
	4		3		4		
	4						

From Table 3, we can obtain the domain values $V^1 = \{3, 4, 5, 6, 7\}$ on the attribute A_1. According to the Eqs. (3), (4) and (9), the weight of each value can be calculated as follows:

$$\begin{aligned} \omega(3) &= \frac{1}{4}(\frac{f_1'(3)}{r_1} + \frac{f_2'(3)}{r_2} + \frac{f_3'(3)}{r_3} + \frac{f_4'(3)}{r_4}) \\ &= \frac{1}{4}(\frac{2}{5} + \frac{2}{4} + \frac{1}{4} + \frac{1}{2}) \\ &= \frac{1.65}{4}, \end{aligned}$$

In the same way, we have $\omega(4) = \frac{0.90}{4}$, $\omega(5) = \frac{0.45}{4}$, $\omega(6) = \frac{0.5}{4}$, $\omega(7) = \frac{0.5}{4}$. Clearly, $\omega(3) > \omega(4) > \omega(6) = \omega(7) > \omega(5)$. And $u_1 = round(\frac{3+3+4+2}{4}) = 3$, so we choose 3, 4, 6 as values of Q_{A_1}. Furthermore, we can compute $\sum_{i=1}^{4} f_i'(3) = 6$, $\sum_{i=1}^{4} f_i'(4) = 4$, $\sum_{i=1}^{4} f_i'(6) = 2$. Therefore, the center of **X** can be represented as ((3, 6), (4, 4), (6, 2)).

4. Experiments on real data

In this section, we mainly make some experiments on the five real data sets, Microsoft Web data, Market Basket data, Alibaba data, Musk data and Movielens data, to evaluate the effectiveness of the proposed algorithm. We firstly describe the preprocessing process of the five data sets. Then five evaluation indexes are introduced. Finally, we show the comparison results of the k-mw-modes algorithm with other algorithms and discuss the impact of the parameter ε on the clustering performance.

4.1. Data preprocessing

To our best knowledge, open matrix-object data with label information are very rare. To tackle this problem, we need to conduct data preprocessing for the given real data sets, because the data sets clustered by clustering algorithms are generally supposed to exist some structure or distribution.

To obtain the structure of a given matrix-object data set, we use the multidimensional scaling technique [5] to visualize the data. The main goal of the technique is to obtain a configuration of n points (rows) in P dimensions (cols) by passing the n-by-n distance matrix obtained by Eq. (1) to the function *mdscale* from *MATLAB*. The Euclidean distances between n points approximate a monotonic transformation of the corresponding dissimilarities in the n-by-n distance matrix. Therefore, we can visualize n points to reflect the distribution of the data. To visualize the data, we set $P = 2$.

In most cases, the distribution of a real data set is often disordered. By the visualization technology, we can delete some points to get the relative clear structure of the data. From the visual figure, we can intuitively find the number of clusters and obtain the label information of every matrix-object. Thus, we can use external evaluation indexes to evaluate the clustering performance of the k-mw-modes algorithm. Below is the preprocessing process of the five real data sets.

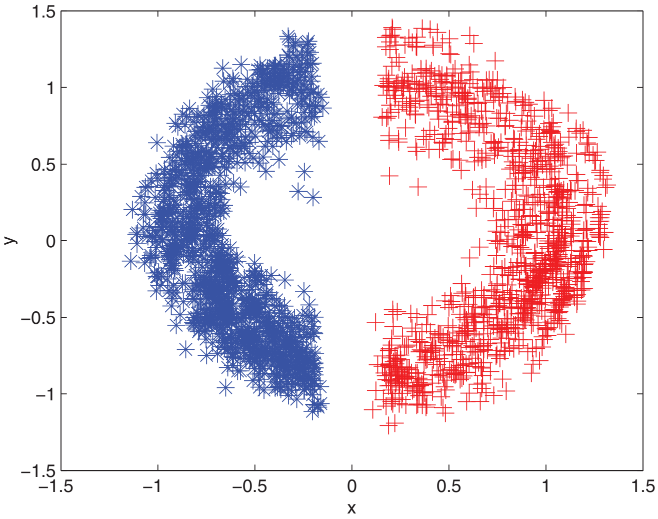

Fig. 1. The distribution of Microsoft Web data after preprocessing.

4.1.1. Microsoft Web data

Microsoft Web data set downloaded from UCI is created by sampling and processing the http://www.microsoft.com logs. It records the visiting of the web sites in one week timeframe in February 1998 by 32,711 anonymous, randomly-selected users. For each user, it is described by two attributes, User_Id and Web_Id, and more than one web site are visited. So, each user is a matrix-object and the data set can be used to evaluate the k-mw-modes algorithm.

The preprocessing process is as follows: firstly eliminate the objects that visit less than seven web sites to generate a temporary set; secondly visualize the temporary set in the coordinate system [5] and select the objects whose abscissa values are in the position of $x < -0.1$ or $x > 0.1$ to form a new set of 2401 objects; finally visualize the new set shown in Fig. 1.

Obviously, Microsoft Web data set can be divided into 2 clusters.

4.1.2. Market Basket data

Market Basket data downloaded from Data website[1] record the transactions of 1001 customers, each of which is described by four attributes, Customer_Id, Time, Product_Name and Product_Id. Here, we just need to take Customer_Id and Product_Id into account and ignore the other attributes, because all customers have the same values on the attribute Time and Product_Name can be replaced completely by Product_Id. In addition, the prominent characteristic for Market Basket data is that every customer in it has 7 transactional records. Therefore, each customer is a typical matrix-object.

Following is the process of Market Basket data preprocessing. By visualizing the data with multidimensional scaling technique, we select some objects who locate the position of $x < -0.2$, $y < 0.5$ or $x > 0$, $y < -0.1$ or $x > -0.3$, $y > 0.7$ or $x > 0.3$, $y > 0.1$ in the coordinate system to form a new data set of 900 objects. The distribution of the new data set is shown in Fig. 2.

It is clearly that Market Basket data can be divided into 3 clusters.

4.1.3. Alibaba data

Alibaba data downloaded from the competition website[2] describe user's behavior of visiting brands. It records 182,880 visiting records of 884 users who are described by four attributes, User_Id, Time, Action_type and Brand_Id. In this experiment, we

[1] http://www.datatang.com/datares/go.aspx?dataid=613168.
[2] http://102.alibaba.com/competition/addDiscovery/index.htm.

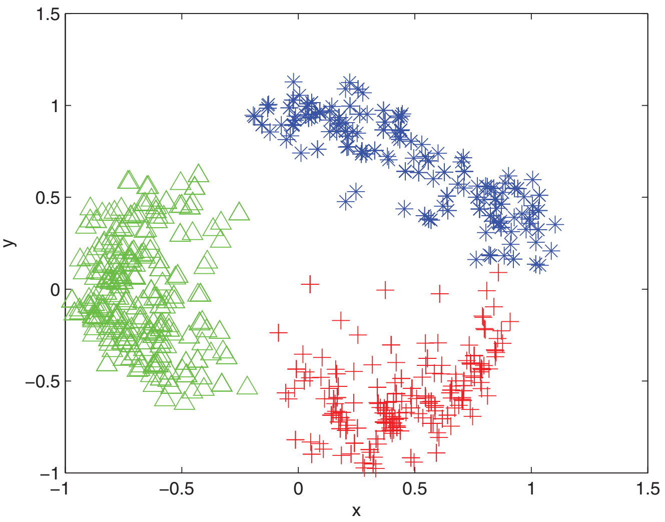

Fig. 2. The distribution of Market Basket data after preprocessing.

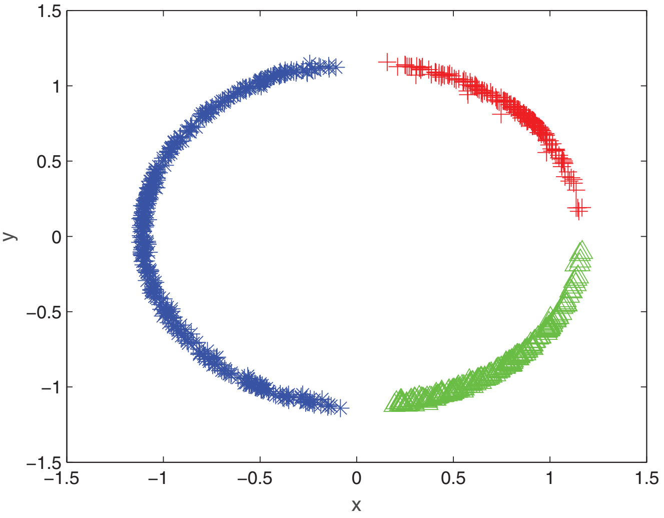

Fig. 3. The distribution of Alibaba data after preprocessing.

only consider the attribute User_Id and Brand_Id. We can see that Alibaba data set is a matrix-object data set because every user visited more than one brand.

The preprocessing process of Alibaba data is described as follows: firstly visualize the data in the coordinate system; secondly eliminate the objects whose abscissa values are in the range of $-0.2 < x < 0.2$ or $x > 0.2$, $0 < y < 0.2$ to obtain a new set of 793 objects; finally visualize the new set in Fig. 3.

Clearly, Alibaba data are divided into 3 clusters according to Fig. 3.

4.1.4. Musk data

Musk data describe a set of 92 objects, each of which represents a molecule and is described by 167 attributes. This data set can be downloaded from UCI [6] and aims at predicting whether new molecules will be musks or non-musks. The fact that 476 instances or records are contained in Musk data results in more than one record for an object. That is to say, Musk data set is a matrix-object data set. Furthermore, it has been divided into 2 clusters by the human experts that the 47 molecules are judged to be musks while the remaining 45 molecules are non-musks. In this experiment, we consider attribute values on Musk data as categorical values. Therefore, we can cluster Musk data directly without preprocessing.

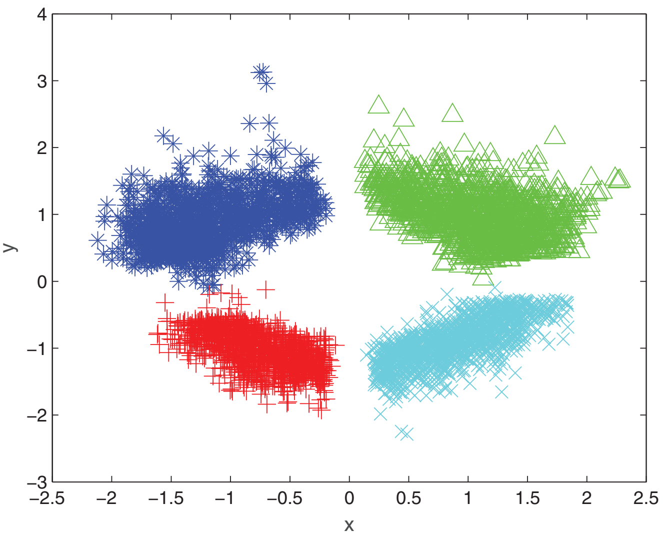

Fig. 4. The distribution of MovieLens data after preprocsssing.

Table 4
Data sets after preprocessing.

Data set	Matrix-objects	Attributes	Records	k
Web	2401	2	22,250	2
Basket	900	2	6300	3
Alibaba	793	2	165,655	3
Musk	92	167	476	2
Movie	4592	3	728,487	4

4.1.5. MovieLens data

MovieLens data are collected and made from the MovieLens website[3] and they contain three different size parts, MovieLens 100k, MovieLens 1M and MovieLens 10M. In this experiment, we make some experiments on the MovieLens 1M data, which contain three files including movies data, ratings data and users data. Movies data and users data are not considered because they only describe the fundamental information of movies and users respectively.

In this experiment we only employ the ratings data set which records 1,000,209 ratings of 3900 movies made by 6040 users selected randomly. And it is described by four attributes, UserID, MovieID, Rating and Timestamp. The attribute Timestamp has a different value for every record. We delete the attribute because it has almost no effects on clustering results.

The preprocessing process is described as follows: select objects that are in the range of $x < -0.2$, $y > 0.4$ or $x > 0.2$, $y > 0.4$ or $x < -0.2$, $y < -0.5$ or $x > 0.2$, $y < -0.3$ in the coordinate system as a new set after the visualization of the initial data set; then visualize the new data in Fig. 4.

Apparently, the data seen from Fig. 4 are divided into 4 clusters.

The final data sets after preprocessing are listed in Table 4. These data sets are used to evaluate the k-mw-modes algorithm.

4.2. Evaluation indexes

To evaluate the effectiveness of the k-mw-modes clustering algorithm, we used the following five external criterions: (1) adjusted rand index (ARI) [7], (2) normalized mutual information (NMI) [8], (3) accuracy (AC), (4) precision (PE) and (5) recall (RE) to

[3] http://grouplens.org/datasets/movielens/.

Table 5
The contingency table.

	C_1	$C_2 \ldots\ldots$	$\cdots$	$C_{k'}$	*Sums*
P_1	n_{11}	n_{12}	$\cdots$	$n_{1k'}$	p_1
P_2	n_{21}	n_{22}	$\cdots$	$n_{2k'}$	p_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
P_k	n_{k1}	n_{k2}	$\cdots$	$n_{kk'}$	p_k
Sums	c_1	c_2	$\cdots$	$c_{k'}$	n

measure the similarity between two partitions of objects in a given data set.

Let **X** be a matrix-object data set, $C = \{C_1, C_2, \ldots, C_k'\}$ be a clustering result of **X**, $P = \{P_1, P_2, \ldots, P_k\}$ be a real partition in **X**. The overlap between C and P can be summarized in a contingency table shown in Table 5, where n_{ij} denotes the number of objects in common between P_i and C_j, $n_{ij} = |P_i \cap C_j|$. p_i and c_j are the number of objects in P_i and C_j, respectively.

The five evaluation indexes are defined as follows:

$$\mathrm{ARI} = \frac{\sum_{ij} C^2_{n_{ij}} - [\sum_i C^2_{p_i} \sum_j C^2_{c_j}]/C^2_n}{\frac{1}{2}[\sum_i C^2_{p_i} + \sum_j C^2_{c_j}] - [\sum_i C^2_{p_i} \sum_j C^2_{c_j}]/C^2_n},$$

$$\mathrm{NMI} = \frac{\sum_{i=1}^{k} \sum_{j=1}^{k'} n_{ij} \log(n_{ij} n / p_i c_j)}{\sqrt{\sum_{i=1}^{k} p_i \log(p_i/n) \sum_{j=1}^{k'} c_j \log(c_j/n)}},$$

$$\mathrm{AC} = \frac{1}{n} \max_{j_1 j_2 \ldots j_k \in S} \sum_{i=1}^{k} n_{ij_i},$$

$$\mathrm{PE} = \frac{1}{k} \sum_{i=1}^{k} \frac{n_{ij_i^*}}{p_i},$$

$$\mathrm{RE} = \frac{1}{k'} \sum_{i=1}^{k'} \frac{n_{ij_i^*}}{c_i},$$

where $n_{1j_1^*} + n_{2j_2^*} + \cdots + n_{kj_k^*} = \max_{j_1 j_2 \ldots j_k \in S} \sum_{i=1}^{k} n_{ij_i}$ $(j_1^* j_2^* \ldots j_k^* \in S)$ and $S = \{j_1 j_2 \ldots j_k : j_1, j_2, \ldots, j_k \in \{1, 2, \ldots, k\}, j_i \neq j_t \text{ for } i \neq t\}$ is a set of all permutations of 1, 2, . . ., k. For *AC*, *PE*, *RE*, k is equal to k' in general case. In addition, we consider that the higher the values of *ARI*, *NMI*, *AC*, *PE* and *RE* are, the better the clustering solution is.

4.3. Comparisons of two cluster center update algorithms GAFMWM and HAFMWM

In this section, we show comparison results of two cluster center update algorithms *GAFMWM* and *HAFMWM* used in the k-mw-modes algorithm. As the k-mw-modes algorithm that is the extension of the k-modes algorithm is sensitive to the initial cluster centers, we executed the k-mw-modes algorithm with two cluster center update algorithms 10 times, respectively. And all of our experiments were conducted on a PC with an Intel Xeon CPU I7(3.4 GHz) and 16 GB memory. Table 6 shows the average values and standard deviations of clustering evaluation indexes on Market Basket data set.

Table 7
Run-time of the k-mw-modes algorithm with *GAFMWM* and *HAFMWM* on Market Basket data.

	Run-time (Second)
k-mw-modes + *GAFMWM*	$1.60497 \times 10^5 \pm 0.4731 \times 10^5$
k-mw-modes + *HAFMWM*	15.6146 ± 4.2214

In addition, we compared the execution time of the k-mw-modes algorithm with two update methods and the experimental results are shown in Table 7.

From Table 6, we can find that the k-mw-modes algorithm with *GAFMWM* is better than it with *HAFMWM* on all evaluation indexes. But we also can see that the k-mw-modes algorithm with *GAFMWM* is very time consuming from Table 7. It took about 45 h to produce one clustering result on Market Basket data, which is not acceptable in real applications. However, the k-mw-modes algorithm with *HAFMWM* only took a few seconds to produce a clustering result on the same data set. *HAFMWM* compared with *GAFMWM* speeds up the k-mw-modes process tremendously. Furthermore, the values of evaluation indexes of both algorithms are closer. Therefore, we will use the k-mw-modes algorithm with *HAFMWM* instead of *GAFMWM* in the following experiments.

4.4. Comparisons of the k-mw-modes with other algorithms

As far as we know, no appropriate algorithms can be used to directly cluster the categorical matrix-object data. To cluster the data by some existed algorithms, we have to convert the representation of matrix-objects.

If we use a feature vector to represent a matrix-object, the matrix-object data can be clustered by k-modes-type algorithms. We choose the attribute value with maximal frequency as a representative on each attribute for each matrix-object. Then, each matrix-object is simplified a feature vector. In this subsection, we compared the k-mw-modes with the k-modes, the *Wk*-modes [16], the k-modes with the new dissimilarity measure (abbr. k-modes-cao) [10].

If we regard all categories to be independent, the matrix-object data can be identified with co-occurrence information data among objects and categories. Thus, some co-clustering algorithms [20–22] can be used to cluster the matrix-object data. For a given matrix-object data set, we count the number of each attribute value in each object to transform each object into a p-dimensional feature vector ($p = \sum_{s=1}^{m} |V^s|$). In this subsection, we compared the k-mw-modes with FCCM that is one of co-clustering methods [20].

In this experiment, we clustered every data set 50 times respectively by the these algorithms and ε was set to 0.1 in the k-mw-modes algorithm. In the FCCM, we set $T_u = 0.1, T_w = 1.5$ and $\epsilon = 0.0001$ [20]. The results of the five data sets are listed in Tables 8–12, respectively.

The left of the symbol "±" in these tables represents the means of external criterion values for 50 experiments and the right of it represents standard deviations. From the five tables, we can find that the means on AC, PE, RE, ARI, NMI in the k-mw-modes algorithm are generally higher than that in the k-modes, the *Wk*-modes and the k-modes-cao algorithms. What's more, the k-mw-modes algorithm on the index AC is approximately 10–30% more than those algorithms on it except for Musk data. We can also see that the AC

Table 6
Comparison results of the k-mw-modes algorithm with *GAFMWM* and *HAFMWM* on Market Basket data.

	AC	PE	RE	ARI	NMI
k-mw-modes + *GAFMWM*	0.9058 ± 0.1176	0.9072 ± 0.1151	0.9004 ± 0.1262	0.8015 ± 0.2215	0.8099 ± 0.2022
k-mw-modes + *HAFMWM*	0.8834 ± 0.1370	0.8906 ± 0.1385	0.8726 ± 0.1483	0.7383 ± 0.2419	0.7180 ± 0.2162

Table 8
Comparison results of the five algorithms on Microsoft Web data.

Algorithms	AC	PE	RE	ARI	NMI
k-modes	0.6117 ± 0.0608	0.6101 ± 0.0598	0.5827 ± 0.0802	0.0592 ± 0.0613	0.0414 ± 0.0432
Wk-modes	0.6284 ± 0.0742	0.7158 ± 0.0615	0.6009 ± 0.1002	0.0806 ± 0.0876	0.0852 ± 0.0786
k-modes-cao	0.6504 ± 0.0653	0.6521 ± 0.0690	0.6357 ± 0.0789	0.1048 ± 0.0835	0.0790 ± 0.0654
FCCM	—	—	—	—	—
k-MW-modes	0.9220 ± 0.0940	0.9288 ± 0.0962	0.9148 ± 0.1013	0.7463 ± 0.1918	0.6806 ± 0.1745

Table 9
Comparison results of the five algorithms on Market Basket data.

Algorithms	AC	PE	RE	ARI	NMI
k-modes	0.4939 ± 0.0670	0.4967 ± 0.0723	0.4865 ± 0.0883	0.0836 ± 0.0564	0.0821 ± 0.0535
Wk-modes	0.5213 ± 0.0643	0.6375 ± 0.0815	0.5240 ± 0.1004	0.1095 ± 0.0615	0.1620 ± 0.0638
k-modes-cao	0.5478 ± 0.0605	0.5806 ± 0.0763	0.5449 ± 0.0738	0.1373 ± 0.0520	0.1517 ± 0.0529
FCCM	0.5405 ± 0.0211	0.5629 ± 0.0458	0.6711 ± 0.1167	0.1522 ± 0.0205	0.1690 ± 0.0233
k-MW-modes	0.8834 ± 0.1370	0.8906 ± 0.1385	0.8726 ± 0.1483	0.7383 ± 0.2419	0.7180 ± 0.2162

Table 10
Comparison results of the five algorithms on Alibaba data.

Algorithms	AC	PE	RE	ARI	NMI
k-modes	0.4817 ± 0.0000	0.4817 ± 0.0012	0.3333 ± 0.0000	0.0000 ± 0.0020	0.0022 ± 0.0015
Wk-modes	0.4832 ± 0.0011	0.7796 ± 0.0821	0.3353 ± 0.0015	−0.0012 ± 0.0040	0.0217 ± 0.0085
k-modes-cao	0.4817 ± 0.0000	0.4805 ± 0.0024	0.3333 ± 0.0000	0.0036 ± 0.0040	0.0049 ± 0.0030
FCCM	—	—	—	—	—
k-MW-modes	0.6528 ± 0.0459	0.6217 ± 0.0564	0.5840 ± 0.0525	0.2866 ± 0.0732	0.2245 ± 0.0610

Table 11
Comparison results of the five algorithms on Musk data.

Algorithms	AC	PE	RE	ARI	NMI
k-modes	0.5567 ± 0.0382	0.5639 ± 0.0434	0.5547 ± 0.0398	0.0089 ± 0.0239	0.0155 ± 0.0192
Wk-modes	0.5159 ± 0.0070	0.6141 ± 0.0988	0.5070 ± 0.0107	−0.0010 ± 0.0014	0.0182 ± 0.0231
k-modes-cao	0.5387 ± 0.0259	0.5402 ± 0.0278	0.5361 ± 0.0295	−0.0021 ± 0.0111	0.0067 ± 0.0086
FCCM	0.5846 ± 0.0285	0.5976 ± 0.0398	0.5820 ± 0.0272	0.0224 ± 0.0206	0.0285 ± 0.0200
k-MW-modes	0.5600 ± 0.0413	0.5688 ± 0.0487	0.5565 ± 0.0423	0.0116 ± 0.0276	0.0183 ± 0.0243

Table 12
Comparison results of the five algorithms on MovieLens data.

Algorithms	AC	PE	RE	ARI	NMI
k-modes	0.4885 ± 0.1206	0.5191 ± 0.1508	0.4864 ± 0.1194	0.1997 ± 0.1205	0.2360 ± 0.1416
Wk-modes	0.5267 ± 0.1220	0.6147 ± 0.1235	0.5351 ± 0.1248	0.2617 ± 0.1231	0.3251 ± 0.1474
k-modes-cao	0.4348 ± 0.0809	0.4266 ± 0.0733	0.4311 ± 0.0794	0.1301 ± 0.0809	0.1261 ± 0.0754
FCCM	—	—	—	—	—
k-MW-modes	0.6244 ± 0.0303	0.6851 ± 0.0297	0.6233 ± 0.0321	0.3211 ± 0.0424	0.3904 ± 0.0367

in the *Wk*-modes and *k*-modes-cao algorithms is generally higher than it in the *k*-modes algorithm.

For the FCCM, its accuracy on Musk data is higher than that of the proposed algorithm, but it has not clustering results on Alibaba data and MovieLens data. The reason is that we cannot compute the memberships of attribute values for each cluster after normalization because some attribute values have a fairly high frequency in each object and the membership of every attribute value in a cluster defined in the FCCM increases exponentially as the increasing of the total frequency of the value in each object. For Microsoft Web data, we only can obtain a cluster, which is meaningless in real applications. Therefore, the FCCM is not suitable for the clustering of this type of data.

In short, it can be seen that the *k*-mw-modes algorithm is exactly better than the other four algorithms.

4.5. Impact of ε

In the *k*-mw-modes algorithm, the parameter ε is used to determine whether the algorithm stops or not. How to decide the size of ε is a very difficult problem, because the clustering results may have some differences when different values of ε are selected. To analyze the impact of ε on the clustering performance of the *k*-mw-modes algorithm, we ran the algorithm 30 times with different values of ε, from 0.02 to 0.2 with step 0.01, on the five data sets and recorded the means of accuracy (AC) and iterations. Results of AC and iterations on the five data sets are shown in Figs. 5 and 6, respectively.

From Fig. 5, we can observe that the AC on Musk, Alibaba and MovienLens has almost no change and the AC on the other two data has been fluctuating as the increasing of ε, but overall they are relatively stable. Meanwhile, from Fig. 6, we can see clearly that the iterations on the five data all show a decreasing trend on the whole as the increasing of ε. Particularly, for Web and Alibaba, the iterations decrease rapidly when $\varepsilon < 0.1$ and they decrease slowly relatively when $\varepsilon > 0.1$. Higher AC and fewer iterations are expected in this algorithm, so we set $\varepsilon = 0.1$.

5. Related work

In real applications, categorical data are widespread. The *k*-modes algorithm [3] extends the *k*-means algorithm [4] by using a simple matching dissimilarity measure for categorical objects,

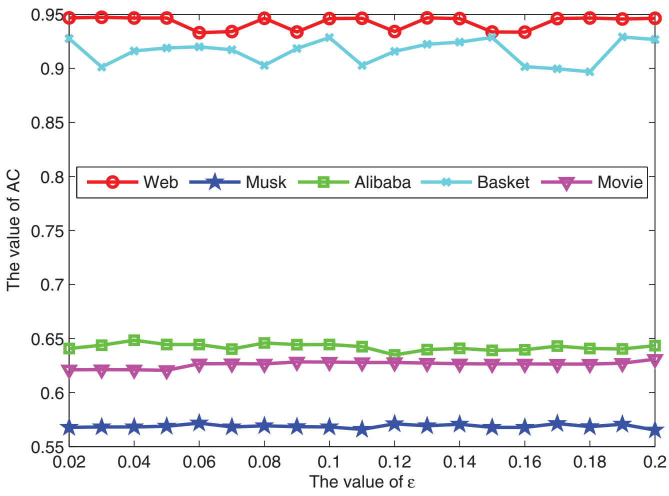

Fig. 5. The AC of the five data with different ε.

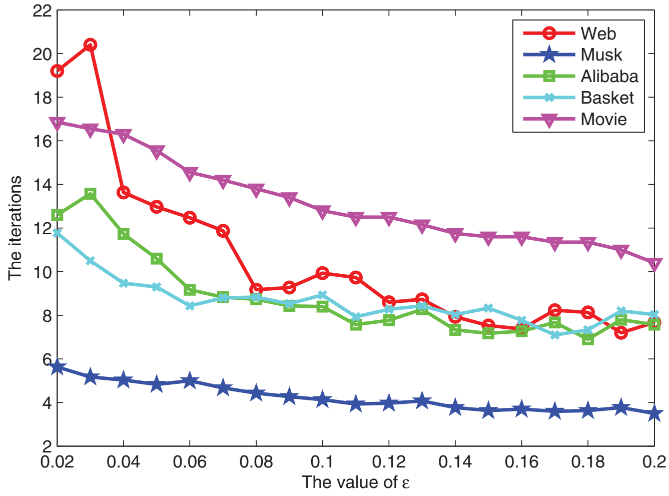

Fig. 6. The iterations of the five data with different ε.

modes instead of means for clusters, and a frequency-based method to update modes in the clustering process to minimize the clustering objective function. These extensions have removed the numeric-only limitation of the *k*-means algorithm and enable the *k*-means clustering process to be used to efficiently cluster large categorical data sets from real world [9,10]. The *k*-modes clustering algorithm [11,12] was also proposed independently. Huang [13] gave the relationship of the two *k*-modes methods. So far, the *k*-modes algorithm and its variants [14–16], including the fuzzy *k*-modes algorithm [17], the fuzzy *k*-modes algorithm with fuzzy centroid [18], the *k*-prototype algorithm [3] and the w-*k*-means [19] have been used widely in many domains. However, these methods can not cluster matrix-object data set effectively.

6. Conclusions

In many database applications, the behavioral traits of a customer are carried in a detail table instead of a master table. To find the customer groups with different behavioral traits, a *k*-mw-modes algorithm was proposed for clustering categorical matrix-object data. In the proposed algorithm, the distance between two matrix-objects was defined and the representation and update ways of cluster centers were developed further. The convergence of the proposed algorithm was proved and the corresponding time complexity was analyzed as well. To speed up the clustering process, a heuristic method was proposed to construct multi-weighted-modes centers in each iteration of the *k*-mw-modes algorithm. Experimental results on the five real data sets have shown that the *k*-mw-modes algorithm is better than the *k*-modes-type algorithms and the FCCM in clustering categorical matrix-object data.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (under grants 61573229, 61473194, 61432011 and U1435212), the Natural Science Foundation of Shanxi Province (under grant 2015011048), the Shanxi Scholarship Council of China (under grant 2016-003) and the National Key Basic Research and Development Program of China (973) (under grant 2013CB329404).

References

[1] J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques, 3rd ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011.
[2] R. Xu, D. Wunsch, Clustering, vol. 10, John Wiley & Sons, 2008.
[3] Z. Huang, Extensions to the *k*-means algorithm for clustering large data sets with categorical values, Data Mining Knowl. Discov. 2 (3) (1998) 283–304.
[4] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, California, USA, 1967, pp. 281–297.
[5] S. Schiffman, L. Reynolds, F. Young, Introduction to Multidimensional Scaling: Theory, Methods, and Applications, Academic Press, 1981.
[6] K. Bache, M. Lichman, UCI Machine Learning Repository, 2014 http://archive.ics.uci.edu/ml.
[7] J. Liang, L. Bai, C. Dang, F. Cao, The *k*-means type algorithms versus imbalanced data distributions, IEEE Trans. Fuzzy Syst. 20 (4) (2012) 728–745.
[8] A. Strehl, J. Ghosh, Cluster ensembles – a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (2003) 583–617.
[9] F. Cao, J.Z. Huang, J. Liang, Trend analysis of categorical data streams with a concept change method, Inform. Sci. 276 (2014) 160–173.
[10] F. Cao, J. Liang, D. Li, L. Bai, C. Dang, A dissimilarity measure for the *k*-modes clustering algorithm, Knowl.-Based Syst. 26 (2012) 120–127.
[11] J.D. Carroll, A. Chaturvedi, P.E. Green, *k*-Means, *k*-Medians and *k*-Modes: Special Cases of Partitioning Multiway Data, 1994.
[12] A. Chaturvedi, P.E. Green, J.D. Caroll, *k*-modes clustering, J. Classif. 18 (1) (2001) 35–55.
[13] Z. Huang, M.K. Ng, A note on k-modes clustering, J. Classif. 20 (2) (2003) 257–261.
[14] M.K. Ng, M.J. Li, J.Z. Huang, Z. He, On the impact of dissimilarity measure in *k*-modes clustering algorithm, IEEE Trans. Pattern Anal. Mach. Intell. 29 (3) (2007) 503–507.
[15] L. Bai, J. Liang, C. Dang, F. Cao, The impact of cluster representatives on the convergence of the *k*-modes type clustering, IEEE Trans. Pattern Anal. Mach. Intell. 35 (6) (2013) 1509–1522.
[16] F. Cao, J. Liang, D. Li, X. Zhao, A weighting k-modes algorithm for subspace clustering of categorical data, Neurocomputing 108 (2013) 23–30.
[17] Z. Huang, M.K. Ng, A fuzzy *k*-modes algorithm for clustering categorical data, IEEE Trans. Fuzzy Syst. 7 (4) (1999) 446–452.
[18] D.-W. Kim, K.H. Lee, D. Lee, Fuzzy clustering of categorical data using fuzzy centroids, Pattern Recogn. Lett. 25 (11) (2004) 1263–1271.
[19] J.Z. Huang, M.K. Ng, H. Rong, Z. Li, Automated variable weighting in *k*-means type clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (5) (2005) 657–668.
[20] C.-H. Oh, K. Honda, H. Ichihashi, Fuzzy clustering for categorical multivariate data, in: Proceedings of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference, 2001, pp. 2154–2159.
[21] K. Kummamuru, A. Dhawale, R. Krishnapuram, Fuzzy co-clustering of documents and keywords, Proceedings of the 2003 IEEE International Conference on Fuzzy Systems 2 (2003) 772–777.
[22] H. Frigui, O. Nasraoui, Simultaneous categorization of text documents and identification of cluster dependent keywords, Proceedings of the 2002 IEEE International Conference on Fuzzy Systems 2 (2002) 1108–1113.